Сравнительный анализ непараметрических алгоритмов на примере моделирования стохастических процессов

Е. Д. Михов[1,*], Е. Д. Михова[2], О. В. Непомнящий[1]

[1]Сибирский федеральный университет, Красноярск, Россия

[2]Сибирский государственный аэрокосмический университет, Красноярск, Россия

[*]Контактный e-mail: edmihov@mail.ru

Рассмотрена проблема моделирования стохастических безынерционных процессов в пространстве входных-выходных переменных. Указаны некоторые отличия между непараметрическими алгоритмами моделирования, а именно моделирования при помощи ядерной аппроксимации и персептрона Розенблатта — Парзена. Описаны принципы работы нейросетей и алгоритма обучения нейросетей. Раскрыто строение используемого в исследованиях персептрона. Рассмотрен алгоритм моделирования при помощи ядерной аппроксимации. Продемонстрирован результат оптимизации вектора "коэффициентов размытости ядра". Оптимизация проводилась при помощи алгоритма Недлера — Мидда.

Приведены результаты моделирования при помощи персептрона Розенблатта и ядерной аппроксимации. Показано, что между выбранными алгоритмами построения моделей нет существенного различия в точности.

Ключевые слова: моделирование стохастических процессов, ядерная аппроксимация, нейросеть, сравнение алгоритмов.

Введение

В связи с повсеместным внедрением компьютерно-вычислительных систем появляются все новые и новые способы моделирования производственных процессов, и у исследователя может возникнуть закономерный вопрос: в каких условиях тот или иной алгоритм моделирования более предпочтителен?

В настоящей статье рассмотрены два подхода математического моделирования: при помощи персептрона Розенблатта — Парзена [1] и алгоритма ядерного сглаживания (ядерной аппроксимации) [2], построены модели стохастических безынерционных процессов. На рис. 1 приведена стандартная схема исследуемого процесса. Несмотря на кажущуюся простоту изучаемого процесса, при моделировании могут возникать сложности. Например, элементы вектора $\boldsymbol{\xi}(t)$ могут быть слишком высокими для построения качественной модели. Также возникают трудности из-за малой размерности выборки x_i, u_i, $i = \overline{1,s}$, где s — объем выборки. Кроме того, сам объект может быть сложным, что также затрудняет построение модели. Для полноты исследования в статье продемонстрированы результаты построения математической модели объекта каждым из алгоритмов в разных условиях.

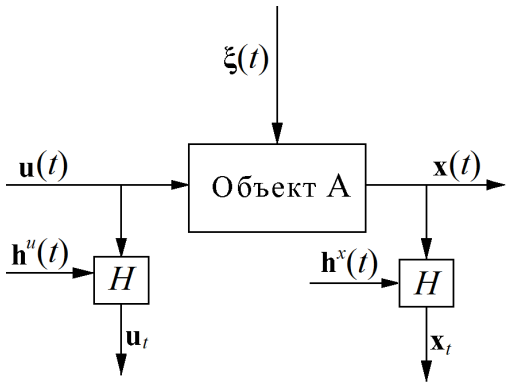

Рис. 1. Общая схема исследуемого процесса: A — исследуемый объект (процесс), $\mathbf{x}(t)$ — выходной вектор процесса, $\mathbf{u}(t)$ — входной вектор процесса, $\boldsymbol{\xi}(t)$ — случайное воздействие (помеха)

1. Идентификация в “узком” и “широком” смысле

При моделировании разнообразных дискретно-непрерывных процессов доминирует теория идентификации в “узком” смысле. Суть ее состоит в том, что на первом этапе на основании имеющейся априорной информации определяется параметрический класс оператора объекта A^α, например:

$$\hat{x}_\alpha(t) = A^\alpha(u(t), \alpha), \tag{1}$$

где A^α — параметрическая структура модели, $\boldsymbol{\alpha}$ — вектор параметров. На втором этапе осуществляется оценка параметров $\boldsymbol{\alpha}$ на основе имеющейся выборки x_i, u_i, $i = \overline{1,s}$, s — объем выборки. Успех решения задачи идентификации в этом случае существенно зависит от того, насколько “удачно” определен оператор (1).

Идентификация в “широком” смысле предполагает отсутствие этапа выбора параметрического класса оператора. Часто оказывается значительно проще определить класс операторов на основе сведений качественного характера, например линейности процесса или типа нелинейности, однозначности либо неоднозначности и др. В этом случае задача идентификации состоит в оценивании этого оператора на основе выборки x_i, u_i, $i = \overline{1,s}$. При идентификации в “широком” смысле хорошее качество выборки имеет особую значимость. Под качеством здесь подразумевается и точность снимаемых данных, и равномерность распределения измерений по вектору $\mathbf{u}(t)$. Качество данных важно, так как без параметрического класса оператора оценка будет осуществляться только по ним:

$$\hat{x_s}(t) = A_s(u(t), \mathbf{x}_s, \mathbf{u}_s),$$

где $\mathbf{x}_s = (x_1, x_2, \ldots, x_s), \mathbf{u}_s = (u_1, u_2, \ldots, u_s)$ — временны́е векторы. Оценка оператора A_s может быть осуществлена средствами непараметрической статистики [3]. Примечательно, что при этом исключается этап выбора параметрической структуры. Таким образом, можно утверждать, что идентификация в этом случае, а это вариант идентификации в “широком” смысле, более адекватна реальным задачам практики.

2. Персептрон Розенблатта

Одной из первых нейронный сетей, способных к обучению, является персептрон Розенблатта, который рассматривался им не как конкретное техническое вычислительное

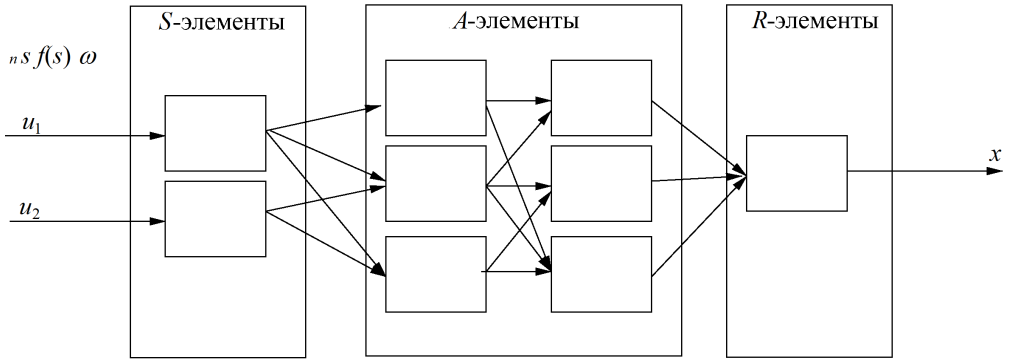

Рис. 2. Элементарный персептрон Розенблатта

устройство, а как модель работы мозга. Самый простой вид персептрона представлен на рис. 2

Персептрон содержит элементы трех типов: входной вектор $\mathbf{u}_s$ поступает на S-элементы, которые в дальнейшем передают его в нейросеть. Далее $\mathbf{u}_s$ поступает в слой ассоциативных или A-элементов. Ассоциативные элементы представляют собой слой нейронов, они выполняют нелинейную обработку информации и имеют изменяемые веса связей. R-элементы с фиксированными весами формируют сигнал реакции персептрона на входной стимул. Однослойный персептрон характеризуется матрицей весов связей W от S- к A-элементам. Элемент матрицы W_{ij} отвечает весу связи, ведущей от i-го S-элемента к j-му A-элементу.

В работах Розенблатта сделано заключение о том, что нейронная сеть рассмотренной архитектуры способна к воспроизведению любой функции, однако, как было показано позднее М. Минским и С. Пейпертом [4], этот вывод оказался неточным. Были выявлены принципиальные неустранимые ограничения однослойных персептронов, и впоследствии в основном стал рассматриваться многослойный вариант персептрона, в котором имеются несколько слоев процессорных элементов.

3. Искусственный нейрон

Нейрон состоит из умножителей (синапсов), сумматора и нелинейного преобразователя. Синапсы осуществляют связь между нейронами, умножают входной сигнал на число, характеризующее силу связи (вес синапса). Сумматор выполняет сложение сигналов, поступающих по синаптическим связям от других нейронов и внешних входных сигналов. Нелинейный преобразователь реализует нелинейную функцию одного аргумента — выхода сумматора. На рис. 3 показана его структура. Функция f называется функцией активации нейрона. Нейрон реализует скалярную функцию векторного аргумента. Математическая модель нейрона имеет вид

$$x = f(s), \tag{2}$$

$$s = \sum_{i=1}^{n} (w_i u_i),$$

где w_i — вес синапса; s — результат суммирования; u_i — компонент входного вектора (входной сигнал), $i = \underline{1, n}$; x — выходной сигнал нейрона; n — число входов нейрона. Выход x определяется видом функции активации и может быть как действительным, так

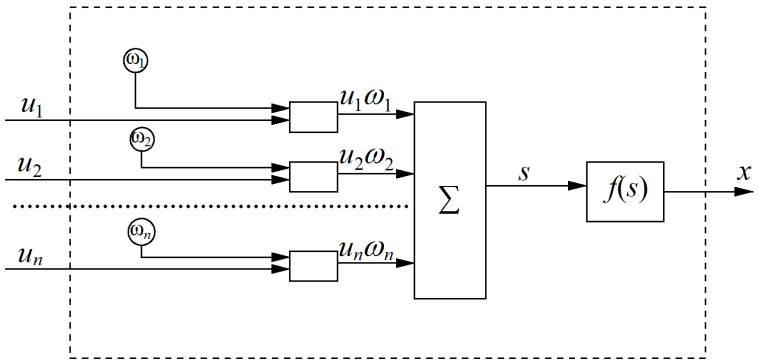

Рис. 3. Структура нейрона

и целым. Синаптические связи с положительными весами называют возбуждающими, с отрицательными весами — тормозящими.

4. Функции активации

Одна из наиболее распространенных функций активации — нелинейная функция активации с насыщением, так называемая логистическая функция или сигмоид (функция S-образного вида):

$$f(s) = \frac{1}{1 + e^{-as}}. \tag{3}$$

При уменьшении a сигмоид становится более пологим, в пределе при $a = 0$ вырождаясь в горизонтальную линию на уровне 0.5, при увеличении a сигмоид приближается к виду функции единичного скачка с порогом T. Из выражения (3) очевидно, что выходное значение нейрона лежит в диапазоне $(0, 1)$. Одно из ценных свойств сигмоидальной функции — простое выражение для ее производной (4):

$$f'(s) = af(s)(1 - f(x)).$$

Сигмоидальная функция дифференцируема на всей оси абсцисс, что используется в некоторых алгоритмах обучения. Кроме того, она обладает свойством усиливать слабые сигналы лучше, чем большие, и предотвращает насыщение от больших сигналов, так как они соответствуют областям аргументов, где сигмоид имеет пологий наклон.

5. Метод обратного распространения ошибки

Рассмотрим процесс обучения нейронной сети с использованием алгоритма обратного распространения ошибки. Этот метод заключается в итерационной поправке весовых коэффициентов синапсов (связей между нейронами). Продемонстрируем, каким образом вычисляются корректировки весов на каждой итерации.

Воспользуемся дельта-правилом:

$$\delta w_{jk} = h\delta_k x_j,$$

где h — норма обучения, число, которое мы сами задаем перед началом обучения; x_j — это выход нейрона j, поступающий ко входу нейрона k; δ_k — ошибка нейрона k.

Таким образом, в процессе обучения на вход сети подаются элементы обучающей выборки, и в результате получаются новые значения весовых коэффициентов. Обычно обучение заканчивается, когда для всех вводимых образцов величина ошибки станет меньше определенного значения. После этого сеть подвергается тестированию при помощи новых данных, которые не участвовали в обучении. По результатам этого тестирования можно сделать выводы, хорошо или нет справляется сеть со своими задачами.

Определим ошибку

$$\delta = y - \hat{y},$$

где y — выходная переменная исследуемого объекта, $\hat{y}$ — выходная переменная нейросети.

Алгоритм определяет два "потока" в сети. Входные сигналы двигаются в прямом направлении, в результате чего получается выходной сигнал, из которого мы имеем значение ошибки. Величина ошибки двигается в обратном направлении, в итоге происходит корректировка весовых коэффициентов связей сети.

Для корректировки весовых значений используем дельта-правило. Необходимо определить универсальное правило для вычисления ошибки каждого элемента сети после прохождения через элемент (при обратном распространении ошибок). Данное правило выглядит следующим образом:

$$\delta_j = f'(y_j) \sum_{k=1}^{p} \delta_k w_{jk}.$$

Здесь y_j — выход нейрона j.

6. Демонстрация работы нейросети

Рассмотрим работу нейросетей на примере. Создадим нейросеть, имеющую архитектуру, представленную на рис. 4.

В качестве функции активации возьмем функцию (3). Математическая модель данного персептрона выглядит следующим образом:

$$x = \frac{1}{1 + \exp^{\omega_{35} u_3 + \omega_{45} u_4}}, \tag{4}$$

где ω_{ij} — вес на синапсе между нейронами под номерами i и j, а u_i — сигнал (выходное значение) из нейрона под номером i,

$$u_3 = \frac{1}{1 + \exp^{\omega_{13} u_1 + \omega_{23} u_2}}, \tag{5}$$

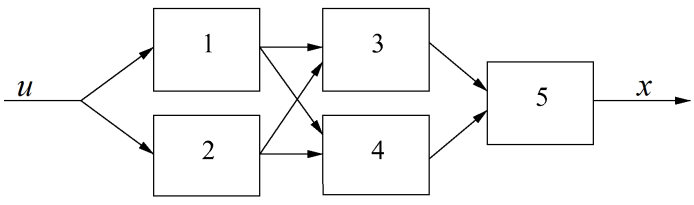

Рис. 4. Структура созданного персептрона

$$u_4 = \frac{1}{1 + \exp^{\omega_{14}u_1 + \omega_{24}u_2}}, \tag{6}$$

$$u_1 = \frac{1}{1 + \exp^{\omega_{01}u}}, \tag{7}$$

$$u_2 = \frac{1}{1 + \exp^{\omega_{02}u}}. \tag{8}$$

Здесь ω_{0i} — вес между входом в нейросеть и нейроном под номером i. Объединяя формулы (4)–(8), получаем итоговую математическую модель:

$$x = (1 + \exp(-(\omega_{35}(1 + \exp^{\omega_{13}(1+\exp^{\omega_{01}\cdot u})^{-1} + \omega_{23}(1+\exp^{\omega_{02}u})^{-1}})^{-1} +$$

$$+\omega_{45}(1 + \exp^{\omega_{14}(1+\exp^{\omega_{01}u})^{-1} + \omega_{24}(1+\exp^{\omega_{02}u})^{-1}})^{-1})))^{-1} \tag{9}$$

По формуле (9) минимальное значение модели равно 0, а максимальное 1. Для моделирования исследуемых процессов при помощи данной нейросети необходимо, чтобы минимально возможный выход объекта соответствовал выходу нейросети 0, а максимальный — 1. Для этого проведем преобразование

$$x_r = x(\max - \min) + \min .$$

Здесь x_r — преобразованный выход; max и min — максимальный и минимальный выходы объекта.

Как видно из формулы (9), единственные неизвестные параметры — это веса между нейронами. Поскольку в зависимости от вида моделируемого объекта данные веса меняются, сделаем расчет весов для разных типов объектов.

Рассчитаем веса для процесса, описываемого формулой

$$x = 3u.$$

После обучения нейросети веса на связях будут определены следующим образом:

$$\begin{array}{ll} w_{01} = 0.5042780636, & w_{02} = 0.5041280398, \\ w_{13} = -0.9416053438, & w_{14} = 1.0258435508, \\ w_{23} = -0.9892822461, & w_{24} = 0.8413549838, \\ w_{35} = -35.7932352107, & w_{45} = 9.8605341198. \end{array}$$

Подставив веса в формулу (9), построим график зависимости x от u (рис. 5). Здесь xr — истинный выход процесса, xp — выход модели на основе нейросети. Данный вид персептрона при определенной настройке весов может смоделировать поведение линейного процесса.

Изменим вид объекта

$$x = u^2.$$

После обучения нейросети веса на связях будут определены следующим образом:

$$\begin{array}{ll} w_{01} = -0.4753286045, & w_{02} = 0.4359816899, \\ w_{13} = 32.3167577187, & w_{14} = -3.3419524262, \\ w_{23} = -5.873934891, & w_{24} = 4.1821417097, \\ w_{35} = -15.4626824377, & w_{45} = 17.1219421999. \end{array}$$

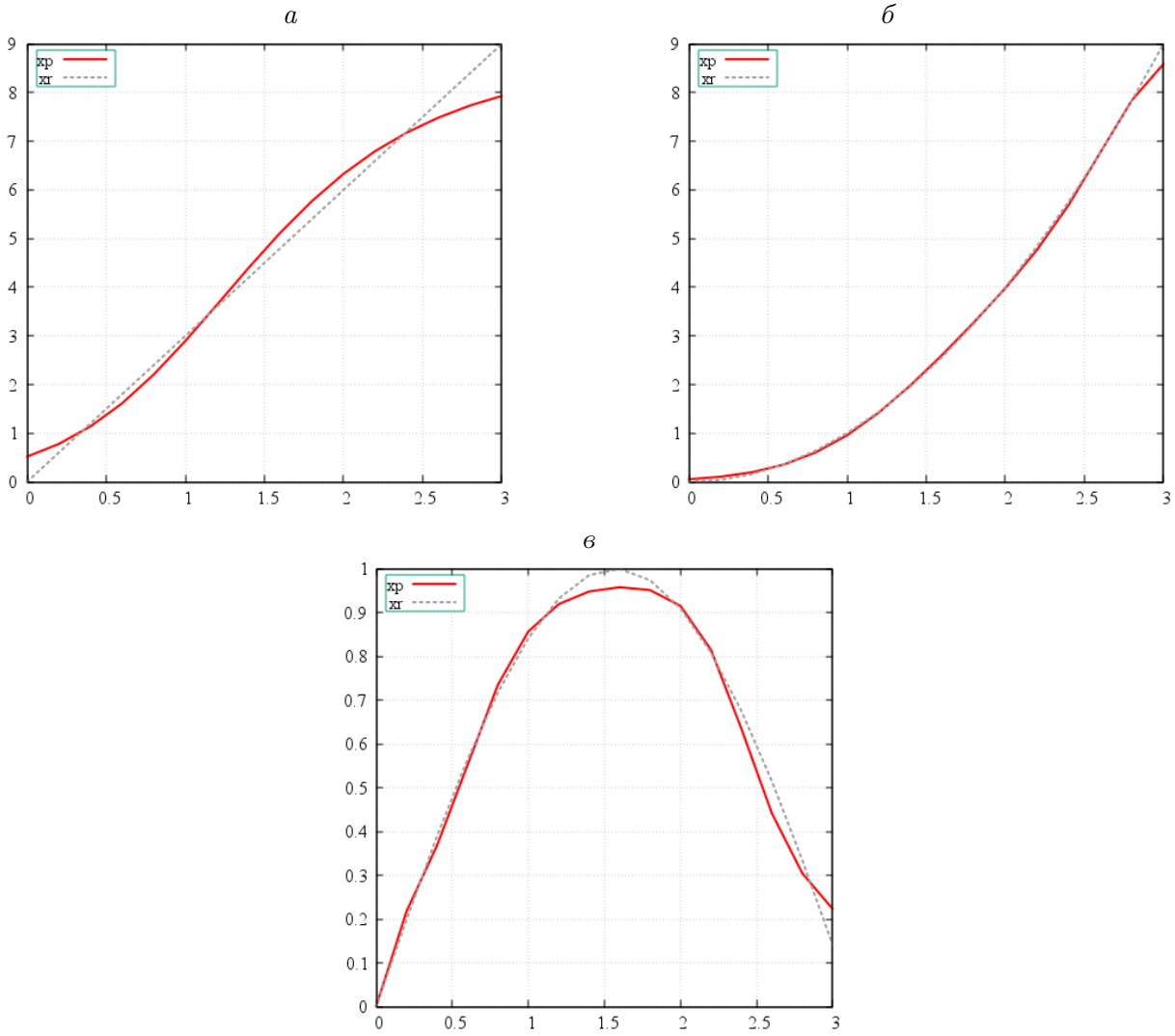

Рис. 5. Модели процессов на основе персептрона, обладающих линейной (*а*), квадратичной (*б*) и синусоидальной (*в*) структурой

Подставив веса в формулу (9), построим график зависимости x от u (рис. 5, *б*). Из рисунка видно, что данный вид персептрона при определенной настройке весов может также смоделировать поведение и квадратичного процесса.

Изменим вид объекта

$$x = \sin(u).$$

После обучения нейросети веса на связях будут определены следующим образом:

$$\begin{aligned} w_{01} &= 1.4913215509, & w_{02} &= 3.1001744036, \\ w_{13} &= -3.5998297354, & w_{14} &= -61.4178433782, \\ w_{23} &= 1.4253734868, & w_{24} &= 60.1059521441, \\ w_{35} &= -40.7276843281, & w_{45} &= 8.7800113792. \end{aligned}$$

Подставив веса в формулу (9), построим график зависимости x от u (рис. 5, *в*). Данный вид персептрона при определенной настройке весов может также смоделировать поведение и синусоидального процесса.

Таким образом, модель на основе нейросети может принимать различные виды, необходима только подстройка весов таким образом, чтобы полученная модель максимально отражала процесс.

7. Ядерная аппроксимация

Ядерная оценка плотности вероятности основывается на методе ядерного сглаживания. Основная идея метода проста: вокруг аппроксимирующей точки строится прямоугольник, каждому элементу выборки назначается вес (чем больше расстояние между элементом выборки и точкой построения прогноза, тем меньше вес), а затем выполняется усреднение взвешенных значений.

В случае, если при ядерном сглаживании используется регрессия нулевого порядка, то взвешенные значения последовательности, которые попали в ядро, усредняются. Ядерная оценка плотности использует те же принципы, что и ядерное сглаживание, однако ее алгоритм имеет некоторые особенности. Обратимся к выражению, определяющему процедуру оценки функции при входной переменной, равной u:

$$\hat{f}_h(u) = \sum_{i=1}^{n} \frac{f_i \Phi((u-u_i)/c_s)}{\Phi((u-u_i)/c_s)}, \tag{10}$$

где f_i — значение функции в точке u_i; c_s — коэффициент сглаживания; Φ — ядерная функция.

Ядерная функция $\Phi(*)$ удовлетворяет следующим условиям:

$$\frac{1}{c_s} \int_{-\infty}^{+\infty} \Phi\left(\frac{t-t_i}{c_s}\right) dt = 1,$$

$$\lim_{c_s \to 0} \frac{1}{c_s} \int_{-\infty}^{+\infty} \varphi(t) \Phi\left(\frac{t-t_i}{c_s}\right) dt = \varphi(t_i).$$

Как следует из выражения (10), плотность в точке u вычисляется как сумма значений ядра для величин, определяемых разностями между значением u и значениями последовательности. При этом точки u, в которых вычисляется плотность, могут и не совпадать со значениями элементов выборки.

8. Результаты и анализ исследований

По результатам исследований работа с нейросетью имеет преимущество перед ядерной аппроксимацией в скорости расчета, в то время как настройка вектора **cs** при ядерной аппроксимации происходит значительно быстрее, чем обучение нейросетей [5].

Одна из особенностей работы с нейросетью — большие затраты времени на обучение. Усложнение объекта и увеличение количества входных переменных влекут за собой увеличение времени обучения. Отсюда возникает закономерный вопрос: при достаточно сложном и многомерном объекте не станет ли обучение нейросети слишком долгим? Поэтому имеет смысл проанализировать зависимость между количеством входных переменных и временем обучения нейросети. Указанная зависимость отображена на рис. 6.

Она имеет квадратичный или экспоненциальный характер. Для уточнения характера зависимости будут проведены дальнейшие исследования. Исходя из рис. 6 при построении модели сложных объектов со множествами входов исследователь может столкнуться с проблемой слишком долгого обучения. Также стоит проанализировать данную зависимость для модели, построенной при помощи ядерной аппроксимации (вместо времени, потраченного на обучение будет взято время настройки вектора **cs**). Зависимость отображена на рис. 6, *б* и имеет линейный характер.

На основании рис. 6 можно предположить, что при моделировании сложных процессов с большим количеством входных переменных настройка вектора **cs** будет происходить значительно быстрее чем обучение нейросети.

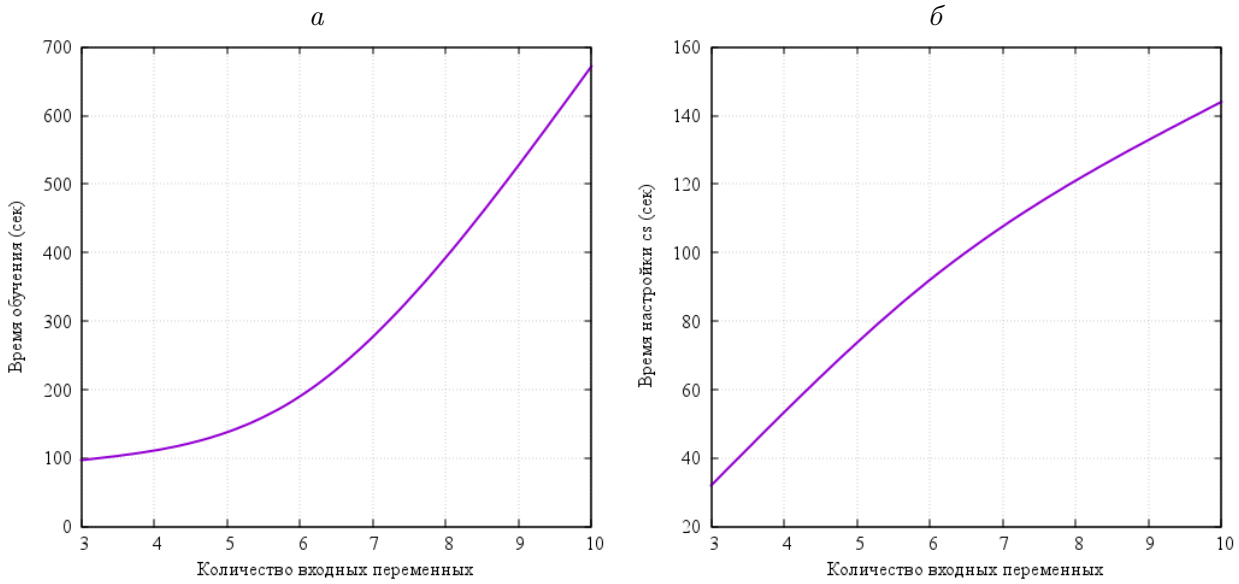

Рис. 6. Зависимость между количеством входных переменных и временем обучения нейросети (*а*) и временем настройки **cs** (*б*)

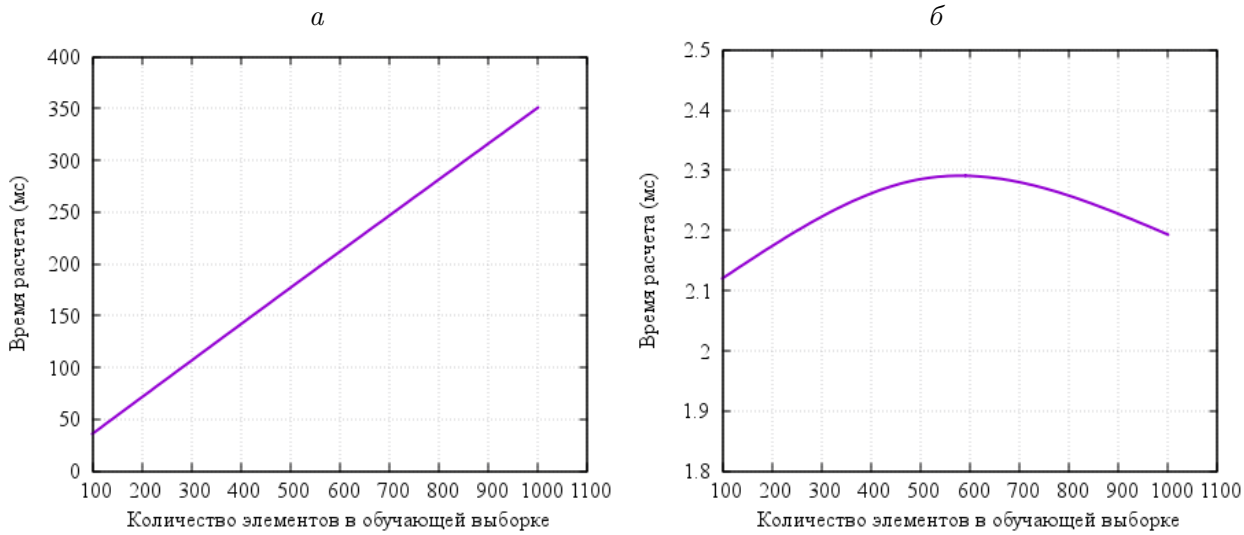

Рис. 7. Зависимость между скоростью вычисления и количеством обучающей выборки в модели, построенной при помощи ядерной аппроксимации (*а*) и нейросети (*б*)

Стоит рассмотреть еще одну особенность, а именно зависимость между скоростью вычисления и количеством обучающей выборки. Связанно это с тем, что существуют процессы, при моделировании которых скорость выдачи прогноза играет существенную роль.

Для начала рассмотрим зависимость между скоростью вычисления и количеством обучающей выборки в модели, построенной при помощи ядерной аппроксимации (рис. 7). Эта зависимость имеет линейный характер. Теперь построим график зависимости между скоростью вычисления и количеством обучающей выборки в модели, построенной при помощи нейросети (рис. 7, *б*). В данном случае между скоростью вычисления и количеством обучающей выборки зависимости нет.

Анализируя рис. 7, можно прийти к выводу, что скорость вычисления для модели, построенной при помощи нейросети, не только выше, чем для модели, построенной при помощи ядерной аппроксимации, но и не зависит от количества обучающей выборки. Это может быть преимуществом при моделировании процессов, где требуется высокая скорость построения прогноза и имеется большое количество обучающей информации.

Таким образом, были продемонстрированы примеры построения моделей двух типов: на основе нейросети и ядерной аппроксимации в различных условиях. Отражены достоинства и недостатки данных подходов. Проведен сравнительный анализ данных моделей.

Благодарности. Работа поддержана Министерством образования и науки. Соглашение 14.578.21.0021. Уникальный идентификатор: RFMEFI57814X0021.

Список литературы / References

[1] **Розенблатт Ф.** Принципы нейродинамики. Перцептроны и теория механизмов мозга. М.: Мир, 1965. 480 с.
Rozenblatt, F. Principles of neurodynamic. Perceptrons and the theory of brain mechanisms. Moscow: Mir, 1965. 480 p. (In Russ.)

[2] **Цыпкин Я.З.** Адаптация и обучение в автоматических системах. М.: Наука, 1968. 400с.
Tsypkin, Ya.Z. Adaptation and learning in automatic systems. Moscow: Nauka, 1968. 400 p. (In Russ.)

[3] **Медведев А.В.** Некоторые замечания к Н-моделям безынерционных процессов с запаздыванием // Вестн. СибГАУ. 2014. № 2(54). С. 50–55.
Medvedev, A.V. Some notes on H-models for non-inertia systems with a delay // Vestnik SibGAU. 2014. No. 2(54). P. 50–55. (In Russ.)

[4] **Минский М., Пейперт С.** Персептроны. М.: Мир, 1971. 261 с.
Minskiy, M., Peypert, S. Perceptrons. Moscow: Mir, 1971. 261 p. (In Russ.)

[5] **Михов Е.Д.** Оптимизация коэффициента размытости ядра в непараметрическом моделировании // Вестн. СибГАУ. 2015. № 2(16). С. 338–342.
Mikhov, E.D. Optimization of coefficient of blurring of the kernel in nonparametric modelling // Vestnik SibGAU. 2015. No. 2(16). P. 338–342. (In Russ.)

Поступила в редакцию 13 июля 2016 г.,
с доработки — 8 ноября 2016 г.

Comparative analysis of nonparametric algorithms on the example of modelling of stochastic processes

Mikhov, Evgeniy D.[1,*], Mikhova, Evgeniya D.[2], Nepomnyashchiy, Oleg V.[1]

[1]Siberian Federal University, Krasnoyarsk, 660041, Russia
[2]Siberian State Aerospace University, Krasnoyarsk, 660037, Russia

*Corresponding author: Mikhov, Evgeniy D., e-mail: edmihov@mail.ru

The problem of modelling of stochastic processes without inertia in the space of input-output variables is considered. The general scheme of the studied processes is described. Some differences between nonparametric algorithms of modelling, namely modelling by means of nuclear approximation and by means of Rosenblat — Parsena's perceptron are considered. The question of identification in "broad" and "narrow" sense is briefly discussed. Distinction between these types of identification is described in detail. The fact that identification in the "broad" sense corresponds more to real problems of modelling rather than identification in "narrow" sense is proved.

Principles of neural networks and it's training algorithms are described. The structure of the perseptron used in research is described.

The modelling algorithm based on the core approximation is considered. The vectors of the "smooth core" optimization result is shown. Optimization was performed by the Nedler — Midd algorithm.

Further results of modelling by means of Rosenblat perceptron and core approximation were presented.

It is shown that between the model designed to use neural network algorithm and the model, that designed to employ core approximation's algorithm, there is no essential difference in accuracy. It is shown that neural networks give the forecast quicker than the method of local approximation, but at the same time neural networks training is much longer.

Keywords: stochastic processes simulating, core approximation, neural networks, comparison of algorithms.

Acknowledgements. The work was supported by the Ministry of Education. Agreement : 14.578.21.0021. Unique identifier : RFMEFI57814X0021.